Singleton Pattern

2019.02.26.

Concept

Shallow Copy:

- The object is created with reference to the address of the parent.
- It will be changed in the value of the parent object.

Deep Copy:

- Copy and create objects completely as separate variables.

Main.java

```java
public class Main{
    public static void main(String[] args){

        Cat navi = new Cat();
        navi.setName("navi");
        navi.setAge(new Age(2012, 3));

        Cat yo = navi.copy();
        yo.setName("yo");//navi name = yo name Same Address value -> Shallow copy : Low level Copy
        yo.setAge(new Age(2013, 2));


        System.out.println(navi.getName());
        System.out.println(yo.getName());

        System.out.println(navi.getAge().getYear());
        System.out.println(yo.getAge().getYear());
    }
}
```

Cat.java

```java
pubic class Cat implements Cloneable{
    private String name;

    public void setName(String name){
        this.name = name;
    }
    public String getName(){
        return name;
    }

    //Deep copy
    public Cat copy() throws CloneNotSupportedException{
        Cat ret = (Cat)this.clone();
        return ret;
    }
}
```

Age.java

```java
public class Age{
    int year;
    int value;
    public Age(int year, int value){
        super();
        this.year = year;
        this.value = value;
    }
    public void setValue(int value){
        this.value = value;
    }
    public void setYear(int year){
        this.year = year;
    }
    public int getValue(){
        return value;
    }
    public int getYear(){
        return year;
    }
}
```

Cat.java

```java
pubic class Cat implements Cloneable{
    private String name;
    private Age age;

    public void setName(String name){
        this.name = name;
    }
    public String getName(){
        return name;
    }
    public Age getAge(){
        return age;
    }
    public void setAge(Age age){
        this.age = age;
    }

    //Deep copy
    public Cat copy() throws CloneNotSupportedException{
        Cat ret = (Cat)this.clone();
        return ret;
    }
}
```

Append Age (get/set)

Cat.java

```java
pubic class Cat implements Cloneable{
    private String name;
    private Age age;

    public void setName(String name){
        this.name = name;
    }
    public String getName(){
        return name;
    }
    public Age getAge(){
        return age;
    }
    public void setAge(Age age){
        this.age = age;
    }

    //Deep copy
    public Cat copy() throws CloneNotSupportedException{
        Cat ret = (Cat)this.clone();
        ret.setAge(new Age(this.age.getYear(), this.age.getValue()));
        return ret;
    }
}
```

→ Explicit copy

Main.java

```java
public class Main{
    public static void main(String[] args){

        Cat navi = new Cat();
        navi.setName("navi");
        navi.setAge(new Age(2012, 3));

        Cat yo = navi.copy();
        yo.setName("yo");
        yo.getAge().setYear(2013);
        yo.getAge().setValue(2);

        System.out.println(navi.getName());
        System.out.println(yo.getName());

        System.out.println(navi.getAge().getYear());
        System.out.println(yo.getAge().getYear());

        Sytem.out.println(navi.getAge().getValue());
        System.out.println(getAge().getValue());
    }
}
```

Case i

Case ii

Result